

## Exercice 1 : Changer le contenu d'un élément

### Énoncé

Créez une page HTML avec un paragraphe contenant du texte par défaut. Ensuite, ajoutez un bouton. Lorsque l'utilisateur clique sur ce bouton, le texte du paragraphe doit être changé par un autre texte.

### Indications

- Utilisez un élément `<p>` pour afficher du texte.
- Utilisez un élément `<button>` pour ajouter un bouton à la page.
- Ajoutez un événement `click` sur le bouton pour modifier le contenu du paragraphe.
- Utilisez la méthode `innerHTML` ou `textContent` pour changer le texte du paragraphe.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Changer le contenu d'un élément</title>
</head>
<body>

  <p id="texte">Texte par défaut.</p>
  <button id="changerTexte">Changer le texte</button>

  <script>
    const paragraphe = document.getElementById('texte');
    const bouton = document.getElementById('changerTexte');

    bouton.addEventListener('click', function() {
      paragraphe.textContent = "Le texte a été changé !";
    });
  </script>

</body>
</html>
```

### Explication

- **HTML** : Le paragraphe contient un texte initial et le bouton déclenche l'événement de modification.
- **JavaScript** : `getElementById` est utilisé pour obtenir les éléments par leur `id`. L'événement `click` du bouton déclenche la modification du texte du paragraphe via `textContent`.

## Exercice 2 : Ajouter un élément à la page

### Énoncé

Créez une page HTML avec un bouton "Ajouter un élément". Lorsqu'on clique sur le bouton, un élément de liste `<li>` avec du texte "Nouvel élément" doit être ajouté à une liste existante.

### Indications

- Créez une liste vide `<ul>` sur la page.
- Utilisez `createElement()` pour créer un élément `<li>`.
- Ajoutez cet élément à la liste avec `appendChild()`.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Ajouter un élément à la page</title>
</head>
<body>

  <ul id="maListe"></ul>
  <button id="ajouterElement">Ajouter un élément</button>

  <script>
    const liste = document.getElementById('maListe');
    const boutonAjouter = document.getElementById('ajouterElement');

    boutonAjouter.addEventListener('click', function() {
      const nouvelElement = document.createElement('li');
      nouvelElement.textContent = "Nouvel élément";
      liste.appendChild(nouvelElement);
    });
  </script>

</body>
</html>
```

### Explication

- **HTML** : La liste est initialement vide. Le bouton sert à ajouter des éléments à cette liste.
- **JavaScript** : On crée un nouvel élément de liste avec `createElement` et on l'ajoute à la liste via `appendChild`.

---

## Exercice 3 : Changer la couleur d'un élément

### Énoncé

Créez une page HTML avec un paragraphe et un bouton. Lorsque l'utilisateur clique sur le bouton, la couleur du texte du paragraphe doit changer.

### Indications

- Utilisez la propriété `style` pour modifier les styles CSS directement avec JavaScript.
- Utilisez `backgroundColor` pour changer la couleur de fond et `color` pour changer la couleur du texte.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Changer la couleur d'un élément</title>
</head>
<body>

  <p id="texte">Ce texte va changer de couleur.</p>
  <button id="changerCouleur">Changer la couleur</button>

  <script>
    const texte = document.getElementById('texte');
    const bouton = document.getElementById('changerCouleur');

    bouton.addEventListener('click', function() {
      texte.style.color = 'red';
      texte.style.backgroundColor = 'yellow';
    });
  </script>

</body>
</html>
```

### Explication

- **HTML** : Le paragraphe est celui dont on va changer les styles, et un bouton déclenche l'action.
- **JavaScript** : On accède à l'élément via `getElementById` et on modifie directement ses styles CSS en utilisant la propriété `style`. `color` change la couleur du texte et `backgroundColor` la couleur de fond.

---

## Exercice 4 : Supprimer un élément

### Énoncé

Créez une page HTML avec un paragraphe et un bouton. Lorsque l'utilisateur clique sur le bouton, le paragraphe doit être supprimé de la page.

### Indications

- Utilisez `remove()` pour supprimer un élément du DOM.
- Accédez au paragraphe par son `id`.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Supprimer un élément</title>
</head>
<body>

  <p id="texte">Ce texte va être supprimé.</p>
  <button id="supprimerTexte">Supprimer le texte</button>

  <script>
    const texte = document.getElementById('texte');
    const bouton = document.getElementById('supprimerTexte');

    bouton.addEventListener('click', function() {
      texte.remove();
    });
  </script>

</body>
</html>
```

### Explication

- **HTML** : Le paragraphe contient du texte, et le bouton déclenche la suppression de cet élément.
- **JavaScript** : On utilise la méthode `remove()` pour retirer l'élément du DOM lorsque le bouton est cliqué.

---

## Exercice 5 : Modifier plusieurs éléments avec une classe

### Énoncé

Créez une page HTML avec plusieurs paragraphes ayant la même classe. Ajoutez un bouton qui, lorsqu'il est cliqué, change la couleur de tous les paragraphes ayant cette classe.

### Indications

- Utilisez `querySelectorAll()` pour sélectionner plusieurs éléments avec la même classe.
- Utilisez `forEach()` pour itérer sur la liste des éléments sélectionnés.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Modifier plusieurs éléments</title>
</head>
```

```
<body>

  <p class="texte">Premier paragraphe</p>
  <p class="texte">Deuxième paragraphe</p>
  <p class="texte">Troisième paragraphe</p>
  <button id="changerCouleurTous">Changer la couleur de tous</button>

  <script>
    const paragraphes = document.querySelectorAll('.texte');
    const bouton = document.getElementById('changerCouleurTous');

    bouton.addEventListener('click', function() {
      paragraphes.forEach(function(paragraphe) {
        paragraphe.style.color = 'blue';
      });
    });
  </script>

</body>
</html>
```

### Explication

- **HTML** : Trois paragraphes partagent la même classe, ce qui permet de les sélectionner ensemble.
- **JavaScript** : `querySelectorAll()` est utilisé pour sélectionner tous les éléments ayant la classe `texte`. On applique une modification de couleur à chacun d'eux en itérant avec `forEach()`.

### Exercice 6 : Modifier l'attribut d'un élément

#### Énoncé

Créez une page HTML avec une image et un bouton. Lorsque l'utilisateur clique sur le bouton, l'attribut `src` de l'image doit être modifié pour afficher une autre image.

#### Indications

- Utilisez un élément `<img>` pour afficher une image.
- Utilisez `getAttribute()` et `setAttribute()` pour modifier l'attribut `src` de l'image.

#### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Modifier l'attribut d'un élément</title>
</head>
<body>

  
  <button id="changerImage">Changer l'image</button>

  <script>
```

```
const image = document.getElementById('image');
const bouton = document.getElementById('changerImage');

bouton.addEventListener('click', function() {
    image.setAttribute('src', 'image2.jpg');
});
</script>
</body>
</html>
```

### Explication

- **HTML** : L'image est affichée avec un attribut `src` pointant vers une première image.
- **JavaScript** : L'attribut `src` est modifié par `setAttribute()`, ce qui permet de changer l'image affichée lorsque l'utilisateur clique sur le bouton.

---

## Exercice 7 : Créer un formulaire dynamique

### Énoncé

Créez un formulaire avec des champs pour le nom et l'email. Lorsque l'utilisateur soumet le formulaire, un nouveau champ pour le numéro de téléphone doit être ajouté au formulaire.

### Indications

- Utilisez `createElement()` pour créer un nouvel élément de champ de formulaire (`<input>`).
- Utilisez `appendChild()` pour ajouter le champ au formulaire existant.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Formulaire dynamique</title>
</head>
<body>

  <form id="formulaire">
    <label for="nom">Nom:</label>
    <input type="text" id="nom" name="nom">
    <br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email">
    <br>
    <button type="button" id="ajouterChamp">Ajouter un champ
téléphone</button>
  </form>

  <script>
```

```
const formulaire = document.getElementById('formulaire');
const boutonAjouter = document.getElementById('ajouterChamp');

boutonAjouter.addEventListener('click', function() {
    const label = document.createElement('label');
    label.setAttribute('for', 'telephone');
    label.textContent = 'Téléphone: ';

    const input = document.createElement('input');
    input.setAttribute('type', 'tel');
    input.setAttribute('id', 'telephone');
    input.setAttribute('name', 'telephone');

    formulaire.appendChild(label);
    formulaire.appendChild(input);
    formulaire.appendChild(document.createElement('br')); //
Ajouter un saut de ligne
});
</script>

</body>
</html>
```

### Explication

- **HTML** : Un formulaire est constitué de deux champs (nom et email) et d'un bouton pour ajouter dynamiquement un champ téléphone.
- **JavaScript** : Lorsque le bouton est cliqué, un label et un champ `input` de type `tel` sont créés et ajoutés au formulaire avec `appendChild()`.

---

## Exercice 8 : Créer une liste de tâches avec possibilité de suppression

### Énoncé

Créez une page avec une liste de tâches. Chaque tâche doit être accompagnée d'un bouton "Supprimer". Lorsque l'utilisateur clique sur le bouton, la tâche correspondante doit être supprimée de la liste.

### Indications

- Utilisez `createElement()` pour créer des éléments de liste (`<li>`).
- Utilisez `removeChild()` pour supprimer l'élément de la liste.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Liste de tâches</title>
</head>
<body>
```

```
<ul id="listeTaches">
  <li>Task 1 <button class="supprimer">Supprimer</button></li>
  <li>Task 2 <button class="supprimer">Supprimer</button></li>
  <li>Task 3 <button class="supprimer">Supprimer</button></li>
</ul>

<script>
  const listeTaches = document.getElementById('listeTaches');

  listeTaches.addEventListener('click', function(event) {
    if (event.target.classList.contains('supprimer')) {
      const tache = event.target.parentElement;
      tache.remove();
    }
  });
</script>

</body>
</html>
```

### Explication

- **HTML** : La liste de tâches est constituée de plusieurs éléments `<li>`, chaque élément ayant un bouton de suppression.
- **JavaScript** : Un gestionnaire d'événements est ajouté à la liste. Lorsque le bouton "Supprimer" est cliqué, la tâche correspondante est supprimée grâce à `remove()`.

---

## Exercice 9 : Ajouter une classe CSS à un élément au clic

### Énoncé

Créez une page avec un élément `<div>`. Lorsque l'utilisateur clique sur ce div, une classe CSS `active` doit être ajoutée à cet élément. Cette classe doit changer la couleur de fond du div.

### Indications

- Utilisez `classList.add()` pour ajouter la classe CSS à l'élément.
- Définissez la classe `active` dans votre fichier CSS.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Ajouter une classe CSS</title>
  <style>
    .active {
      background-color: yellow;
    }
  </style>
```



```
</head>
<body>

  <div id="maDiv" style="width: 200px; height: 100px; border: 1px solid
black;">
    Cliquez ici pour activer la classe
  </div>

  <script>
    const maDiv = document.getElementById('maDiv');

    maDiv.addEventListener('click', function() {
      maDiv.classList.add('active');
    });
  </script>

</body>
</html>
```

### Explication

- **HTML** : Le `div` est un élément sur lequel l'utilisateur peut cliquer pour appliquer un changement de style.
- **CSS** : La classe `active` est définie pour changer la couleur de fond du `div`.
- **JavaScript** : L'événement `click` ajoute la classe `active` à l'élément `div` via `classList.add()`.

---

## Exercice 10 : Faire défiler une page vers le bas

### Énoncé

Créez une page HTML avec un bouton. Lorsque l'utilisateur clique sur le bouton, la page doit défiler jusqu'en bas.

### Indications

- Utilisez `window.scrollTo()` ou `window.scroll()` pour faire défiler la page.
- Utilisez `document.documentElement.scrollHeight` pour obtenir la hauteur totale du document.

### Correction

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Faire défiler la page</title>
</head>
<body>

  <button id="defilerBas">Descendre</button>
```

```
<div style="height: 2000px;">
  <p>Contenu de la page...</p>
</div>

<script>
  const boutonDefiler = document.getElementById('defilerBas');

  boutonDefiler.addEventListener('click', function() {
    window.scrollTo({
      top: document.documentElement.scrollHeight,
      behavior: 'smooth'
    });
  });
</script>

</body>
</html>
```

### Explication

- **HTML** : Un bouton permet de déclencher l'action de défilement de la page.
- **JavaScript** : La méthode `scrollTo()` est utilisée pour faire défiler la page jusqu'à sa hauteur totale, avec un effet de défilement fluide grâce à l'option `behavior: 'smooth'`.

### Application 1

#### Register Participants

Name:

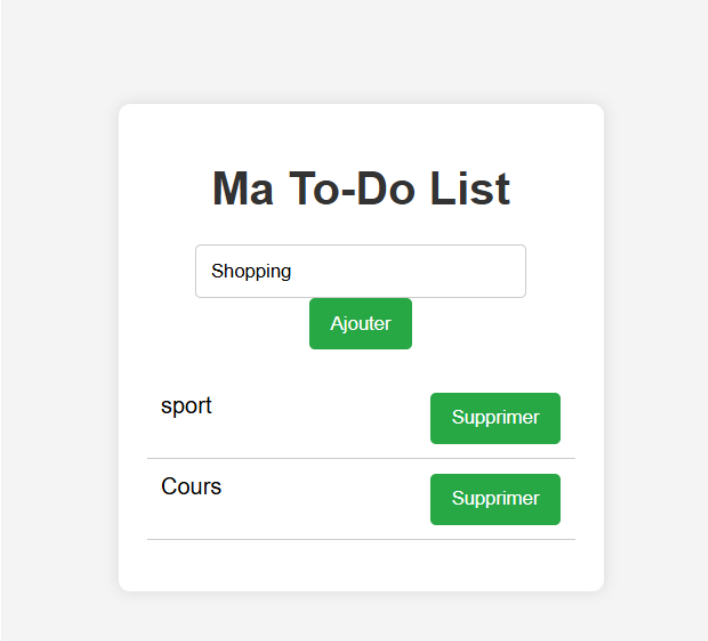
Age:

Email:

#### Participant List

| Name          | Age | Email                   | Action                                |
|---------------|-----|-------------------------|---------------------------------------|
| Kamel ABBASSI | 42  | abbassi.kamel@gmail.com | <input type="button" value="Remove"/> |
| Mohamed amari | 49  | mohamed.amari@gmail.com | <input type="button" value="Remove"/> |

## Application 2



The image shows a web application titled "Ma To-Do List". It features a light gray background with a white rounded rectangle in the center. At the top of the rectangle is the title "Ma To-Do List" in bold black text. Below the title is a text input field containing the word "Shopping". To the right of the input field is a green button with the text "Ajouter". Below the input field and button, there is a list of items. The first item is "sport", followed by a horizontal line, and then a green button labeled "Supprimer". The second item is "Cours", followed by a horizontal line, and then a green button labeled "Supprimer". There is a final horizontal line at the bottom of the list area.

**Ma To-Do List**

Shopping

Ajouter

sport

Supprimer

Cours

Supprimer